# Managed and Continuous Evolution of Dependable Automotive Software Systems*

Andreas Rausch[1]
Oliver Brox[2], Axel Grewe, Marcel Ibe, Stefanie Jauns-Seyfried[2], Christoph Knieke, Marco Körner, Steffen Küpper, Malte Mauritz, Henrik Peters, Arthur Strasser, Martin Vogel, Norbert Weiss[2]

TU-Clausthal, Institute for Applied Software Systems Engineering,
Wallstr. 6, D-38640 Goslar

## Abstract

Automotive software systems are an essential and innovative part of nowadays connected and automated vehicles. Automotive industry is currently facing the challenge to re-invent the automobile. Consequently, automotive software systems, their software systems architecture, and the way we engineer those kinds of software systems are confronted with major challenges: managing complexity, providing flexibility, and guaranteeing dependability of the desired automotive software systems and the corresponding engineering process. In this paper we will present an improved and sophisticated engineering approach. Our approach is based on the managed and continuous evolution of dependable automotive software systems. It helps engineers to manage system complexity based on continous engineering processes to iteratively evolve automotive software systems and therby guarantee the required dependability issues. Based on a running sample, we will present and illustrate the main assets of the proposed engineering approach for managed and continuous evolution of dependable automotive software systems.

## 1. Introduction

The over 125-year-old automotive industry stands for the production of motorised vehicles – motorcycles, passenger cars, commercial vehicles – and with their products for the fulfillment of the human need for mobility. The automotive industry

---

* Der Vortrag wurde am 12.07.2014 vor der Plenarversammlung der Braunschweigischen Wissenschaftlichen Gesellschaft gehalten.

1 Korrespondenzautor

2 Volkswagen AG

is one of the most important industries; in particulary for the German economy with a turnover of 357 billion euros and 750,000 employees in 2012. Moreover, there are engineering firms, car dealers, repair shops, and gas stations[3].

The automotive industry is also top positioned in terms of investments: Between 2002 and 2012, 100 billion euros have been invested in Germany. This corresponds to a share of 23 percent of total industrial investment in Germany. In addition, the automotive industry has invested 77 billion in research and development between 2007 and 2012; almost a third of all expenditures of the German industry for research and development. With its research activities, automotive industry helps to ensure innovation, strengthens the economic, and pushs sustainability in Germany and all over the world.

Since several years, automotive industry is facing the challenge to "re-invent" the automobile (Dieter Zetsche)[4]. Individual mobility is nowadays mainly driven by the following trends: (a) the mobilization of the population in the developing countries promotes the transition from the bike to the car, (b) the differentiated value of cars in the developed countries from a status symbol to a sustainable high-tech mobility service provider, and (c) the common expectations to evolve vehicle towards an integrated, networked piece of the global inter-connected mobility service puzzle.

In the 1970s, the total number of lines of code in vehicle's embedded software was less than 100. 2008, the average in premium vehicles was 10 million lines of code. Software has become an essential part in today's vehicles. Nowadays, 30% to 40% of the added value in the automotive industry is based on software. And, even more important, software is responsible for up to 80% of the innovation in premium vehicles[5].

Thereby, automotive software is not longer restricted to embedded software in vehicles, like control software for the engine or in-vehicle comfort functions. Due to further developments towards the connected car, todays vehicles are equipped with internet access. Hence, the embedded software in vehicles is connected with various Information and Communication Technology (ICT) software and services available via Internet. Consequently, automotive software systems shift towards cyber physical automotive systems based on new system architectures

---

3 cf. http://www.bundesregierung.de/Content/DE/Magazine/emags/economy/051/sp-2-die-automobil-industrie-eine-schluesselindustrie-unseres-landes.html

4 cf. Zetsche, D.: Speech on Reinventing the Automobile Industry in the 21st Century, http://www.worldcarfans.com/104052610585/reinventing-the-automobile-industry-in-the-21st-century

5 cf. eCar-IKT-Systemarchitektur für Elektromobilität: Mehr Software (im) Wagen: Informations- und Kommunikationstechnik (IKT) als Motor der Elektromobilität der Zukunft. http://download.fortiss.org/public/ikt2030/ikt2030de-gesamt.pdf

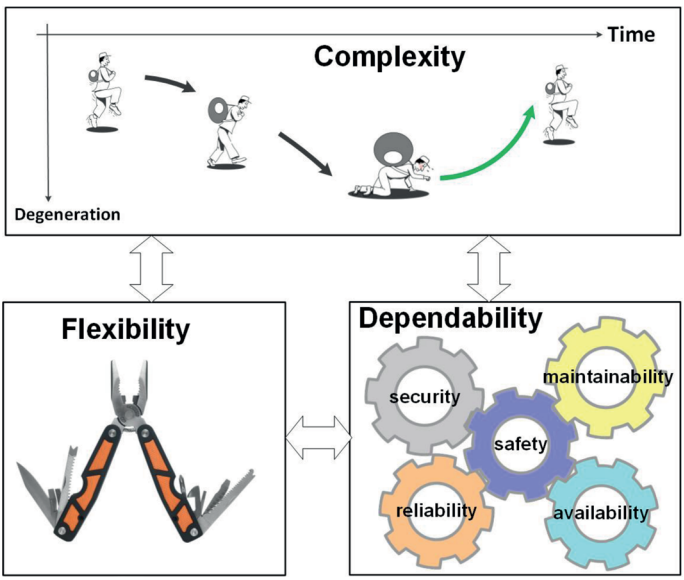


Fig. 1: Three main challenges for automotive software systems engineering.

and development tools that address the complexity and enable the implementation and exploitation of massive amounts of interconnected ICT devices and software services as well as embedded software in physical objects at different locations[6]. To sum up, automotive software systems are an essential and innovative part of nowadays connected and automated vehicles. Automotive industry is currently facing the challenge to re-invent the automobile. Consequently, automotive software systems, their software systems architecture, and the way we engineer those kinds of software systems have to be re-invented. We have identified three main challenges to strengthen automotive software systems engineering for the upcoming (r)evolution, as illustrated in Fig. 1.

a) **Complexity** of automotive software systems and engineering processes has still to be **manageable**.

Usually many variants of a vehicle exist – different configurations of comfort functions, driver assistance systems, connected car services, or powertrains can

6 cf. agendaCPS - Integrierte Forschungsagenda Cyber-Physical Systems. http://www.acatech.de/de/publikationen/empfehlungen/acatech/detail/artikel/acatech-studie-agendacps-integrierte-forschungsagenda-cyber-physical-systems.html

be variably combined, creating an individual and unique product. To keep the vehicles cost efficient, modular components with a high reuse rate cross different types of vehicles are required. With respect to innovative and sophisticated functions, coming with the connected car and automated resp. autonomous driving the functional complexity, the technical complexity, and the networked-caused complexity is continuously and dramatically increasing. It is, and will be in future, a great challenge to further manage the resulting complexity.

b) **Flexibility** of automotive software systems and engineering processes has still to be **provided**.

Developing a new vehicle takes in average about 4 years. Commodity software used for vehicles, like operating systems, multimedia and infotainment software, or network drivers, is updated up to five times faster during vehicle development. This relation is even worse during vehicle operation. Customers expect that new functionality can be easily integrated into vehicles in a plug & play manner. However, nowadays integration of new hardware and software is very expensive. The adaptation of existing components is complex and error-prone. In order to respond quickly to these requirements, the development process must provide a high degree of flexibility.

c) **Dependability** of automotive software systems and engineering processes has still to be **guaranteed**.

Although a high flexible development process and a high reuse rate of automotive software components cross all vehicle types and variants are required, it is crucial to guarantee a high degree of dependability. As dependability, we summarize the essential software quality attributes for vehicles like availability, reliability, maintainability, safety, and security. Due to the connected car and automated resp. autonomous driving, safety and security becomes more and more critical. The actual high warranty costs of about 15% to 20% from earnings before interest and taxes caused by errors in automotive software have to be reduced. Guaranteeing dependability is a great challenge during development and operation of automotive software systems.

To cope with these challenges, we have developed an improved and sophisticated engineering approach for automotive software systems: Our engineering approach is based on the **managed and continuous evolution of dependable automotive software systems**. It helps engineers to manage system complexity based on continuous engineering processes to iteratively evolve automotive software systems, and therby guarantee the required dependability issues.

The rest of the paper is structured as follows: In the next section we present our reseach association Institute for Applied Software Systems Engineering (IPSSE). In section 3, we will describe a running example, helping to illustrate the essential

parts of our engineering approach. The main part of the paper, section 3, presents our engineering approach. A short conclusion and outlook rounds the paper up.

## 2. Research Agenda of Institute for Applied Software Systems Engineering (IPSSE)

The research association Institute for Applied Software Systems Engineering (IPSSE) was found in late 2011 as cooperation between TU Clausthal, TU Braunschweig and Volkswagen AG. The research goal of IPSSE lies in methods and tools for the development of embedded systems. In this scope, one of the prevailing themes of IPSSE is the application of model-driven approaches to automotive software engineering. Therefore we provide a kit containing methods, techniques and tools for successful engineering of embedded software. Our task is to improve this kit with valid and consolidated findings from research, and to transfer it to practice.

Currently, there are five areas of expertise: Reliable reactive systems, adaptive and modular architectures, platform and development tools, hardware/software co-design, and continuing education. IPSSE has a recognized expertise in these fields, on both academic and industrial level.

Concrete approaches in these fields are

- Model-based development with support of product variants, reuse and evolution
- Design of modular architectures
- Measurement and evaluation of architecture erosion and quality with the goal of continuous architecture improvement
- Definition and development of platforms, for example in the multi-core environment
- Model management and automated consistency and quality assurance of models
- Design and implementation of modelling and development tools, e.g. for implementation and testing
- Test-driven development and model-based testing
- Software quality: Formal methods, validation and verification
- Design procedures for real-time systems and distributed realtime systems, also with the aid of co-design
- Transcoding for systems with various design and execution paradigms
- Safety: Detection of execution errors (during runtime) in programs and hardware (in-situ monitoring)

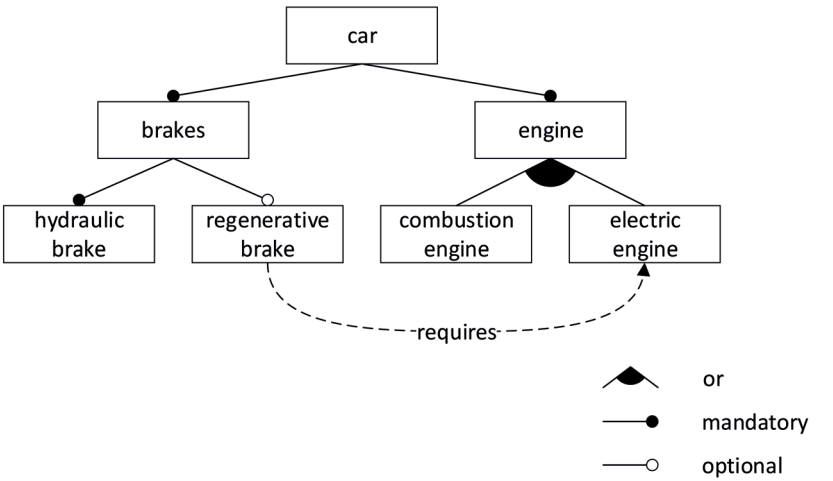


Fig. 2: Feature diagramm (excerpt) of a car.

All research results are demonstrated in demonstrators, prototypes or full-featured tools. Seamless tool support is realised within demonstrating scenarios. The results are validated in the environment of the industrial partners and with their close cooperation[7].

## 3. Running Example

To illustrated the essential parts of the proposed automotive software systems engineering approach we use a small example from the automotive domain: the break system. In the following we will present parts of a break system in an abstract and simplified manner. These aspects will be consequently used in the following sections to illustrate the proposed engineering approach.

The requirements for braking systems became more complex over the years. Conventional hydraulic brakes set the brake pressure, desired by the driver, with the assistance of a pneumatic brake booster. With new engine types, e.g. electrical engine, new brake systems are possible, such as brake energy regeneration for deceleration.

This means the electrical engine is used as a generator to transform kinetic energy into electrical energy; energy is not lost as if using hydraulic brakes.

---

7 see http://www.ipsse.de/

Table 1: Configurations of this example's software product line.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| **Hydraulic brake** | ✓ | ✓ | ✓ | ✓ | ✓ |
| **Regenerative brake** | | | | ✓ | |
| **Combustion engine** | ✓ | | ✓ | | ✓ |
| **Electric engine** | | ✓ | ✓ | ✓ | ✓ |

Existing concepts are not replaced by new ones; they are just enlarged by new technologies.

Corresponding software systems need to support certain features to cover this technical evolution. Thus, the different products can be described as a software product line (SPL) to manage the emerging complexity; a potential feature diagram[8] is shown in Fig. 2.

Even small software product lines can result in a high amount of different configurations. The variaty of configurations for this small example is shown in Table 1. Considering the fact, realising features by various implementations results in an increased number of products.

Changing requirements during the software systems engineering process is demonstrated as well. Therefore, the described software product line is enhanced with the requirements resp. features "cruise control" and "distance sensor", see Fig. 3. This enhancement evolves the architecture and increases the complexity of the architecture.

The integration of these features into the product line represents a major change. Additionally, cruise control is demanded by the customer and needs to be realised soon. Only a high degree of flexibility allows quick response to these new requirements.

For automated driving functionalities high requirement on safety and security do apply; "cruise control" is an example for such functionality. Without guaranteed dependability of the software systems, this feature cannot be realised.

[8] For simplicity reasons, the provided feature model is a reduced excerpt from real existing solutions. For example, regenerative brakes can not only be used in combination with an electric engine.

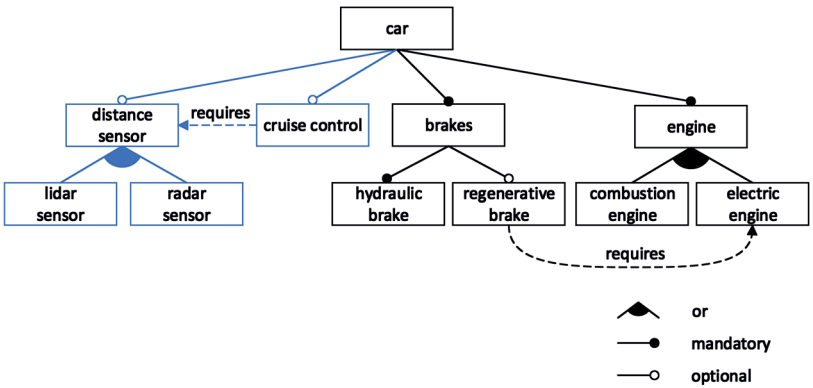


Fig. 3: Extended feature diagramm (excerpt) of a car.

## 4. Approach: Managed and Continuous Evolution of Dependable Automotive Software Systems

As described in Section 1, the connected and automated car comes with three main challenges which are relevant for automotive software systems engineering: **Managing complexity, providing flexibility, and guaranteeing dependability** of the desired automotive software systems and the corresponding engineering process. Therefore, we have developed an improved and sophisticated engineering approach based on the **managed and continuous evolution of dependable automotive software systems**.

The proposed approach helps engineers to **manage functional software systems complexity** based on modular, well-defined, and linked requirements as well as architectures (see Fig. 4). The goal is to create solid requirements and adaequate architectures with the help of abstract principles, patterns, and describing techniques. In addition, we describe how a degenerated architecture can regenerate.

Based on a modular and inter-connected modelling approach for requirements and architectures, we have elaborated an **agile and structured software systems development approach**. This development approach integrates modern agile software development principles with the well established systems development approach based on the classical V-Model. Our approach bases on four driving factors: Systems engineering and agile function development, feature- and function-driven team development, agile management principles, and a seamless tooling infrastructure supporting continously and iteratively evolving automotive software systems.

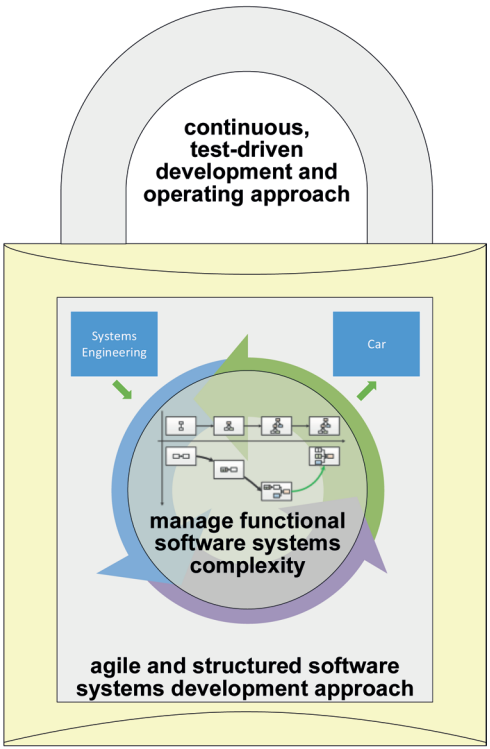


Fig. 4: Managed, continuous evolution of dependable automotive software systems.

And the third element of the proposed approach is a **continuous, test-driven development and operating approach** to guarantee dependability. Thereby, we cover the complete quality assurance lifecycle; from creating test cases to test-driven development during development time as well as during operation time.

## 5. Conclusion

Nowadays, automotive software systems are an essential and innovative part of connected and automated vehicles. Automotive industry is currently facing the challenge to re-invent the automobile. Consequently, automotive software systems, their software systems architecture, and the way we engineer those kinds of software systems are confronted with major challenges: managing complexity, providing flexibility, and guaranteeing dependability of the desired automotive software systems and the corresponding engineering process.

In this paper we have presented an improved and sophisticated engineering approach. Our engineering approach is based on the managed and continuous evolution of dependable automotive software systems. It helps engineers to manage system complexity based on continous engineering processes to iteratively evolve automotive software systems, and therby guarantees the required dependability issues.

But so far, not all parts of the presented approach have been applied combined in a single industrial project. Hence, more reseach and empirical studies are required to validate the approach as a whole. This will be done in the near future by our IPSSE group, and corresponding empirical research results will be published.